9. Ramya

Program Name: Ramya.java   Input File: ramya.dat

Ramya's friend Prashant has been researching permutations, specifically calculating the number of unique possible sequences that can be generated from a list of elements. However, Ramya wants to actually generate the list of unique permutations, which she considers much more of a research challenge than just counting how many there could be, as Prashant is doing.

She knows that given three elements A, B and C, there are six different ways these three elements can be listed in order, starting with the original string, and proceeding from there as follows, ABC, ACB, BAC, BCA, CAB, and CBA, a total of six unique sequences.

She also knows that if there are duplicate elements, as in the list A, B, C and C, there are duplicate sequences, which she wants to eliminate. The unique list would be as shown below.

```
    ABCC, ACBC, ACCB, BACC, BCAC, BCCA, CBAC, CBCA, CABC, CACB, CCAB, CCBA
```

From Prashant's research, she also knows that with multiple duplicates in the list, the process becomes even more complicated, and with more and more elements, the list of unique permutations can become quite long. She decides that she will stop generating permutations once the list reaches 30 permutations, even though there could be many more to follow. She needs your help to write the program to generate up to the first 30 unique permutations of a list of elements, in the order pattern demonstrated in all the examples shown on this page, starting with the original sequence.

Input: Several lines of data, each containing a list of elements represented by single uppercase letters, with single space separation, each sequence possibly containing duplicate elements as demonstrated above. There will be no more than 7 elements in each line of data.

Output: The numbered list of unique permutations for each given list, in the exact order pattern and format shown below, each sequence ending with the single line "=====".

Sample input:

```
ABC
ABCC
ABAB
ABCCC
```

Sample output:

```
ABC
ACB
BAC
BCA
CBA
CAB
=====
ABCC
ACBC
ACCB
BACC
BCAC
BCCA
CBAC
CBCA
CABC
CACB
CCAB
CCBA
=====
ABAB
ABBA
AABB
BAAB
BABA
BBAA
=====
ABCCC
ACBCC
ACCBC
ACCCB
BACCC
BCACC
BCCAC
BCCCA
CBACC
CBCAC
CBCCA
CABCC
CACBC
CACCB
CCABC
CCACB
CCBAC
CCBCA
CCCBA
CCCAB
=====
```